POLITECNICO DI TORINO
Repository ISTITUZIONALE

Zero-Transition Serial Encoding for Image Sensors

Original
Zero-Transition Serial Encoding for Image Sensors / JAHIER PAGLIARI, Daniele; Macii, Enrico; Poncino, Massimo. - In: IEEE SENSORS JOURNAL. - ISSN 1530-437X. - ELETTRONICO. - 17:8(2017), pp. 2563-2571. [10.1109/JSEN.2017.2669921]

Availability:
This version is available at: 11583/2674327 since: 2017-06-08T16:49:46Z

Publisher:
IEEE

Published
DOI:10.1109/JSEN.2017.2669921

Terms of use:
openAccess

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

04 May 2023

Zero-Transition Serial Encoding for Image Sensors

Daniele Jahier Pagliari, *Student Member, IEEE,* Enrico Macii, *Fellow, IEEE,* and Massimo Poncino, *Senior Member, IEEE*

***Abstract*—Off-chip serial buses are the most common interfaces between sensors and processing elements in embedded systems. Due to their length, these connections dissipate a large amount of energy, contributing significantly to the total consumption of the system. The error-tolerant feature of many sensor applications can be leveraged to reduce this energy contribution by means of an approximate serial data encoding.**

In this paper we propose one such encoding called Serial T0, particularly effective for image sensors data. By exploiting the correlation among pixels colors, Serial T0 significantly reduces the energy dissipation on a serial connection, with minimal impact on image quality, and extremely low codec overheads. Moreover, it allows runtime quality reconfiguration, and can be integrated with standard serial protocols (SPI, I2C, CSI-2, etc.) without requiring modifications to legacy peripherals.

We show that Serial T0 is superior to both accurate and approximate state-of-the-art encodings, not only in terms of pure energy saving, but also when combined with realistic error tolerant applications (OCR and face recognition). In the best case, the energy on the bus can be reduced of more than 75% with less than 4% maximum error on the decoded data.

I. Introduction

Serial buses are the most common interface between processing elements and sensor ICs in modern embedded systems. This is due to a number of advantages with respect to parallel connections, such as reduced skew, jitter and crosstalk, which consequently allow higher transmission frequencies. In the case of sensors, however, serial connections are preferred mostly because of their reduced costs in terms of pin count, wiring area and routing complexity. Common protocols for these connections include I2C, SPI, CAN, CSI, etc. [1].

Off-chip sensor-processor interconnects are implemented at the physical level by means of cables or PCB traces with very large capacitive loads. Since these capacitances do not follow the same scaling trend as semiconductor-based ICs [2], [4], [5], their impact on system energy consumption is becoming increasingly relevant. Considering an ultra low-power processor for distributed sensing applications, such as [3], recent studies have shown that the transmission of a single 12-bit word over an off-chip serial bus can roughly consume the same energy as the execution of a 32-bit instruction [4], [5]. This result is even more impressive when considering that it is not uncommon to have tens of sensors attached to a single processor.

Buses dissipation is mostly dynamic, due to charge and discharge of capacitive loads during transitions between electrical levels. At the logic level, these transitions occur in correspondence of bit toggles in the transmitted word. Thus, as widely explored in the past for parallel buses, the simplest and most effective way to optimize energy consists in *encoding* data to be sent on the bus, in a way that reduces the total number of bit toggles [4]–[10], [12].

This paper extends the work presented at the ACM/EDAC/IEEE Design Automation Conference (DAC) 2016. DOI: 10.1145/2897937.2898089

Most existing encoding algorithms for serial buses are *lossless*, i.e., the transmitted data can be reconstructed exactly by the receiver, provided that there are no channel errors [6]–[10]. However, many sensor-based applications can tolerate some loss of information without a significant impact on output quality [13]. For example, tasks in the domains of recognition, control, etc., exhibit this *error resilience* property due to the nature of the involved computations, which (i) are heuristic and involve randomness, or (ii) produce qualitative and clustered outputs (e.g., a classification). Both these elements naturally reduce the impact of errors in the input data [13].

Error resilience can be exploited to construct a *lossy encoding* that, by introducing controlled errors in the transmission, obtains larger savings compared to accurate solutions. The recent Approximate Computing paradigm formalizes the idea of trading off output quality for some other metric (especially energy efficiency) [14]. While many solutions have been proposed to explore this tradeoff in processing elements, few efforts have focused on sensors and interconnects [4], [5], [12].

This paper presents a novel low power serial bus encoding, called Serial T0, that is particularly effective for *image sensors*, such as CCD and CMOS cameras. The encoding is inspired by the T0 technique for parallel address buses in microprocessors [15], in that both exploit data correlation for power saving. However, the proposed solution has several unique features that are specific to the serial nature of the encoding. The main contributions of our work are the following:

- We present the basic idea of Serial T0 and the corresponding hardware implementation of encoding and decoding circuits. Using realistic data from several public databases, we show that Serial T0 is superior with respect to both accurate and approximate state-of-the-art solutions for image data.
- We discuss the integration of Serial T0 with standard serial bus protocols used in commercial image sensors (e.g., the CSI-2 interface by MIPI).
- We evaluate the effectiveness of Serial T0 for two complex applications involving cameras, i.e., OCR and face recognition, for which we obtain a transition count reduction of about 60% while maintaining accuracy as high as 95%.
- We provide an accurate evaluation of the energy consumed on a serial bus, using realistic data for the cable and transmission parameters and taking into account the tradeoff between the overhead due to the codec and the saving in the interconnect.

The paper is organized as follows: Section II resumes the background and related work on serial bus encodings, in particular

for image data. Section III presents the Serial T0 algorithm and its implementation details, while Section IV discusses its integration with standard serial protocols. Section V describes experimental results and Section VI concludes the paper.

II. Background and Related Work

Off-chip serial connections can be modeled, in first approximation, as purely capacitive channels [4], [6], [8], [10], [12]. Under this model, all power dissipation occurs in correspondence of electrical level changes, i.e., it coincides with the dynamic power, and can be computed as:

$$P_{chan} = P_{dyn} = \alpha C_{tot} V_{swing}^2 f \tag{1}$$

where C_{tot} is the total load capacitance, including line driver, pin and wire, V_{swing} is the voltage swing between electrical levels, and f is the transmission frequency. Finally, $\alpha \in [0, 1]$ is the switching probability factor, that accounts for the probability of a level transition in a given clock cycle.

Most energy optimization approaches for off-chip buses, including ours, focus on reducing α. They attempt to minimize the *transition count* (TC), i.e., the number of logic-level changes between adjacent bits of a word (*intra-word*) and of subsequent words (*inter-word*). This is achieved by means of an encoding algorithm, able to produce *codewords* whose switching probability is smaller than that of unencoded (*raw*) data. Literature on serial data encodings is relatively poor with respect to the one on parallel buses, because of the greater optimization space offered by the latter thanks to a large number of physical lines. However, the interest on serial buses is recently increasing, due to the importance of these connections in modern computing systems.

Low energy serial encodings can broadly be categorized in two groups: *lossless* or *accurate*, and *lossy* or *approximate*. Lossless encodings allow exact data recovery upon reception, in absence of channel errors. Hence, they can be used for all types of I/O, i.e., addresses, data and instructions [6]–[10]. Most of these solutions leverage a priori information on the words to be transmitted, and in particular on their *correlation*. For example, the SILENT encoding sends on the bus the bitwise difference (XOR) between subsequent words [6]. The TC is reduced when subsequent data are strongly positively or negatively correlated, because SILENT codewords are dominated by 0s and 1s, respectively. In presence of uncorrelated data, not only SILENT does not reduce the TC, but actually *increases* it. Several other encodings have been devised to overcome this issue [7]–[9], but they all require the transmission of *redundant* bit(s), whose overhead is unacceptable in serial connections. Indeed, if these bits are transmitted in *parallel*, i.e., on a separate line, routing and area costs increase considerably (much more than in the parallel case). If, on the other hand, they are added as header/trailer bits, the available data bandwidth for a given frequency reduces.

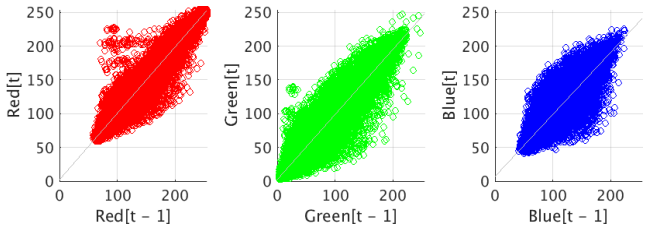
In [10] a lossless bus encoding for *display interfaces* is described. This algorithm exploits data locality in images, hence it can also be used in the context of image sensors. The basic idea is to transmit pixel *differences* rather than pixel values on the bus, based on the strong correlation between adjacent pixels. A look-up-table is used to map the most probable differences to low intra-word TC codewords. A similar approach is proposed in [11] for application in NoCs.

Encodings in the lossy category give up 100% transmission accuracy, in order to obtain larger TC reductions. In other domains (e.g., compression), this concept is very common for multimedia and sensing applications. However, it has been applied directly to serial bus protocols only recently. Needless to say, lossy encodings can only be used for data buses. An early work in the context of image transmission is found in [12], where the authors propose encodings for two of the most popular serial protocols in embedded LCD displays, i.e., DVI and OpenLDI. Both algorithms rely on smart LSB Saturation (LSBS), that is rounding to all zeros or to all ones of some LSBs of the transmitted pixels, also accounting for inter-word transitions. More recently, a new approximate encoding called Rake has been introduced in [4]. Rake is not specifically designed for a particular type of data as it does not rely on correlation assumptions. It is based on heuristically inverting the logic value of some bits within a word, in order to generate long sequences of 1s or 0s, and thus reduce the TC. Inversions are performed under a maximum value-deviation constraint, in order to balance power saving and data fidelity. The algorithm scans each word twice to identify suitable sequences; thus, it has linear complexity in the data bit-width.

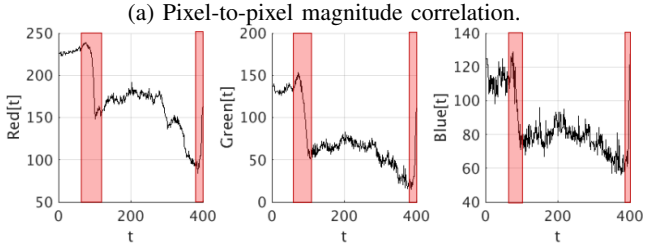
III. Serial T0 Encoding

In this work we present a novel approximate encoding named *Serial T0*, first introduced in [5], specifically designed to optimize power consumption in the data exchange between a processing element (e.g., a System on Chip) and external *sensors*. In particular, this manuscript focuses on using Serial T0 in *image sensors*, i.e., CCD or CMOS cameras. Image sensors are commonly interfaced serially with processing elements via protocols such as *SPI*, *I2C*, and *CSI-2*. Due to the high transmission frequencies (≥ 100 MHz) these connections are often relevant energy consumption contributors [10], [12].

Fortunately, the nature of the serial data *traces* produced by cameras allows to design an encoder that strongly reduces this contribution. In fact, as for several other sensors, serial camera traces exhibit long bursts of high *magnitude correlation* [10]. That is, consecutive pixels transmitted on the bus tend to have similar magnitude, on average, whereas, uncorrelated pixels are an exception. An example is shown in Figure 1a, where each scatter plot refers to one of the three RGB components of the *Lena* image [24]. The plots relate the color components of the pixels sent on the bus at time $t-1$ and at time t, assuming that the matrix is sliced by rows. Correlation coefficients are ≥ 0.91 for all three color channels (p-value < 0.05). Highly correlated word sequences are interrupted by brief phases of large magnitude variation (low correlation). This is exemplified in Figure 1b, which shows initial portion of the bus trace produced by Lena, under the assumption that each color component is transmitted on a separate serial line. Two high correlation sequences are present, separated by abrupt variations, highlighted in red. In the image, these correspond to areas of slowly varying color and sharp lines, respectively.



(a) Pixel-to-pixel magnitude correlation.



(b) Pixel traces.

Fig. 1: Image bus traces features.

Intuitively, the majority of the information is transmitted during large variation phases (edges are more relevant than constant shades for intelligibility). Therefore, the basic idea of Serial T0 is *to make approximations only when the sequence of data is varying slowly, and to transmit accurately when variations are larger*.

A. Serial T0 Encoding

The proposed encoding is inspired by the T0 encoding for parallel buses [15]. In its basic form, parallel T0 encodes *addresses* sent by a processor towards the instruction memory with the following scheme:

$$(B^{(t)}, INC^{(t)}) = \begin{cases} (B^{(t-1)}, 1) & \text{if } b^{(t)} = b^{(t-1)} + S \\ (b^{(t)}, 0) & \text{otherwise} \end{cases} \quad (2)$$

where $b^{(t)}$ and $B^{(t)}$ are the input word and corresponding codeword at time t, $INC^{(t)}$ is a redundant bit, and S is a constant power of 2 called *stride*, corresponding to the parallelism of the memory (e.g., $S = 4$ for 32-bit addresses). T0 exploits the fact that the instruction memory is mostly accessed in ascending sequences with fixed stride, occasionally interrupted by jumps (loops, function calls, or interrupts). During such sequences, all lines of the bus become idle, and the additional INC bit is set to 1, letting the receiver autonomously compute the correct address. For an infinite sequence, T0 reduces the transition count to 0.

Serial T0 inherits this "conditional" nature, although the two encodings are radically different: T0 is an *accurate* encoding designed for *parallel address* buses, whereas Serial T0 is *approximate* and deals with *serial data* connections. The similarity lies in the analogy between the correlation of consecutive addresses and that of image pixels. Likewise, breaks in the sequentiality of addresses resemble the occurrence of edges.

The basic principle of Serial T0 consists in zeroing bus transitions during high correlation phases by replacing the word to be transmitted with a special zero-TC pattern. The encoding scheme is:

$$B^{(t)} = \begin{cases} \text{0-TC pattern} & \text{if } \|b^{(t)} - b^{(t')}\| \leq T_h \\ b^{(t)} & \text{otherwise} \end{cases} \quad (3)$$

where T_h is a tunable *maximum error threshold*, and time t' represents the last time in which a data word was directly sent on the bus. Notice the similarity between Eq. 2 and Eq. 3. An important difference is that, in a serial bus, repeating the last word does not eliminate transitions in general. Therefore, an appropriate 0-TC pattern is transmitted instead.

Serial T0 decoding is implemented as:

$$b^{(t)} = \begin{cases} b^{(t')} & \text{if } B^{(t)} = \text{0-TC pattern} \\ B^{(t)} & \text{otherwise} \end{cases} \quad (4)$$

i.e., the received codeword is simply copied to the output, except for the 0-TC pattern. In that case, the decoder assumes as output the value of the last valid (non 0-TC) word received. Approximations occur when the decoder output is different from the transmitted word, that is when:

$$0 < \|b^{(t)} - b^{(t')}\| \leq T_h \quad (5)$$

Increasing T_h produces more approximations, but allows to transmit more words as 0-TC patterns. Hence, tuning this parameter allows to explore the tradeoff between energy consumption and output quality.

B. Selection of 0-TC pattern and Implementation Details

Only two N-bit binary patterns produce 0-TC when serialized: $(00..0)$ and $(11..1)$. In order to minimize the TC, one of the two has to be used as special pattern in Serial T0. However, these binary patterns are generally used to represent valid values; for example, in 24-RGB $(00..0)$ and $(11..1)$ are interpreted as minimum/maximum hue of the corresponding primary color.

This issue can be overcome in one of two ways: (i) adding redundancy to distinguish between the special 0-TC pattern and a real binary pattern with all-zero/all-ones, or (ii) dedicating one of the two patterns to be used *exclusively* as 0-TC pattern. We select the second option because, as discussed in Section II, redundancy in serial buses is critical.

Notice that if a real datum equal to the 0-TC pattern was transmitted on the bus, it would be erroneously interpreted by the decoder as a repetition of the previous valid word. In these cases, the encoder must replace the word with its nearest neighbor; assuming $(11..1)$ as the 0-TC pattern, the encoder must transmit $(11..10)$ in place of a real $(11..1)$. This substitution causes an additional error of relative magnitude $1/FS$, where FS if the full-scale range of data words (255 in 24-bit RGB). Therefore, it is preferable to select a 0-TC pattern that occurs rarely as real datum. Although for image pixels both $(00..0)$ and $(11..1)$ are uncommon and in general equally probable, we choose $(11..1)$ as the 0-TC pattern for consistency with [5].

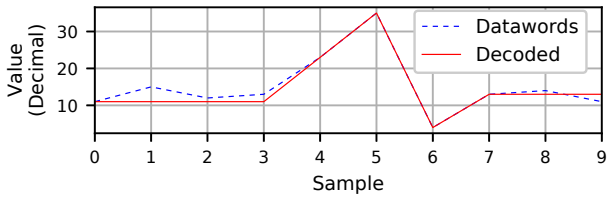


Fig. 2: Example of operation of Serial T0.

Dataword	Total TC	Codeword	Total TC	Decoded	Error
00001011	3	00001011	3	00001011	0
00001111	5	11111111	3	00001011	4
00001100	8	11111111	3	00001011	1
00001101	11	11111111	3	00001011	2
00010111	15	00010111	7	00010011	0
00100011	19	00100011	11	00100011	0
00000100	22	00000100	14	00000100	0
00001101	25	00001101	17	00001101	0
00001110	28	11111111	17	00001101	1
00001011	31	11111111	17	00001101	2

TABLE I: Example of operation of Serial T0, under a maximum relative error constraint of $3.125\%(8/256)$. (see Fig. 2).

C. Example of Operation

An example showing the functionality of Serial T0 is reported in Figure 2 and Table I. The *maximum* error threshold is set to $T_h = 8$, i.e. $\approx 3\%$ of the full-scale, resulting in a total TC reduction of $\frac{31-17}{31} = 45.2\%$. The count considers both intra-word and inter-word transitions, and assumes MSB-first bit ordering. This remarkable reduction is obtained at the expense of just $\frac{1}{255} = 0.4\%$ *average* error on the decoded data (in general, the average error is smaller than T_h). Notice that, in this example, high and low correlation phases are approximately equal in length. However, the former are normally much longer in real image sensor traces. Thus, in real applications, Serial T0 can obtain even larger TC reductions.

D. Encoder and Decoder Implementation

Another advantage of Serial T0 is its simplicity of implementation, either in software and in hardware. Conceptual block diagrams of the encoding and decoding (CODEC) hardware implementations are shown in Figure 3 and 4, assuming that $(11..1)$ is used as 0-TC pattern.

The encoder is composed of six main elements. A N-bit register stores the last word transmitted without approximations $b^{(t')}$, and another is used to store T_h, so that the maximum error constraint can be changed at runtime. A subtractor and a comparator compute whether the absolute value difference between $b^{(t')}$ and the current word $b^{(t)}$ is larger than T_h. An N-to-1 AND gate is used to detect the occurrence of a real $(11..1)$ pattern. Based on comparator and AND gate outputs, a multiplexer forwards to the serializer either the input data word, the 0-TC pattern, or the replacement pattern $(11...10)$.

The decoder circuitry is even simpler. Again, a N-bit register stores the last valid received word. Another N-to-1 AND gate detects the occurrence of a 0-TC pattern, and a multiplexer produces the final output selecting between

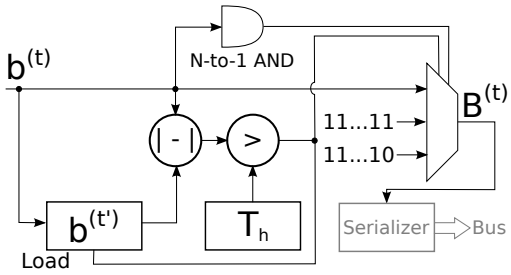


Fig. 3: Serial T0 encoder block diagram.

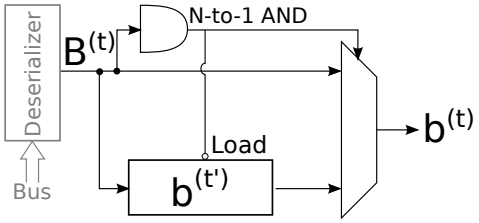


Fig. 4: Serial T0 decoder block diagram.

the current codeword and the last valid word. Notice that the decoding phase is independent from T_h, therefore this parameter does not need to be stored.

IV. Integration with Standard Protocols

In the case of image sensors (cameras), the most widely adopted serial protocols are those developed by the MIPI alliance [16]. Currently, the Camera Serial Interface (CSI-2) is the de facto standard for these sensors, in both prototyping and consumer electronic devices, due to its good trade-off between high speed and cost effectiveness [17]. CSI-2 is commonly combined with a physical level protocol called D-PHY [18], based on *Low Voltage Differential Signaling* (LVDS) pairs, with separate physical *lanes* for data (a maximum of 4) and for a Double Data Rate (DDR), source synchronous clock. In CSI-2, image pixels can be transmitted in multiple formats, including but not limited to RGB, YUV and RAW. Peripheral control and status information, instead, are transmitted on a separate 2-wire physical interface, compatible with the *Inter-Integrated Circuit* (I2C) protocol [19]. A simplified schematic of a CSI-2 over D-PHY interconnect is shown in Figure 5.

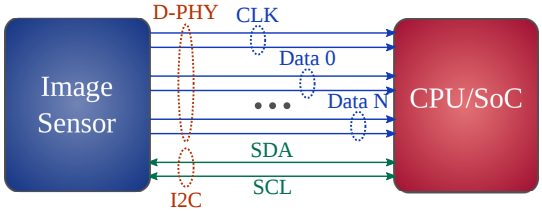


Fig. 5: Camera Serial Interface-2 Interconnect Example.

In order to integrate Serial T0 with CSI-2, care must be taken to *only* encode raw pixel data. In fact, to preserve communication correctness, control bytes (sync-sequences, packet headers and footers, etc.) cannot be approximated. Thus, pixel data must be encoded with Serial T0 *prior* to applying CSI-2. However, since Serial T0 does not alter the semantics of

the data (a pixel component remains a pixel component), the functionality of CSI-2 does not have to be modified. Since LVDS is a level-based encoding, every transition eliminated by Serial T0 on data directly translates to power savings on the bus (the overall impact is proportional to the amount of data words over the total words transmitted). These considerations apply to all protocols that use level-based physical encoding (SPI, I2C, CAN, etc.). Obviously, the net impact of Serial T0 will be higher in simpler protocols (like SPI [20]) with respect to more complex ones (like CAN [21]), due to the different impact of control-related bytes.

When dealing with color images, in principle, the codec of Figure 3 and 4 must be replicated for each color channel, since different color components are not cross-correlated. However, an optimized implementation can reduce costs by sharing hardware (namely the mux. and the N-to-1 AND gate) if the physical data lanes are less than 3. In terms of pure transition count reduction, the optimal interconnect configuration is one that uses a *separate physical wire* for each color component (e.g., three lanes in CSI-2). In fact, this allows to further reduce the power consumption, eliminating also some *inter-word* transitions when multiple 0-TC patterns are transmitted in sequence. In general, however, this advantage has to be balanced with the cost of the encoder, which thanks to sharing slightly reduces for a smaller number of lanes.

A. Advantages over variable bit-width transmission

Importantly, the amount of approximation in Serial T0 can be set at runtime *independently* from the underlying transmission protocol. Indeed, setting the quality simply consists in assigning a value to T_h in (3). Moreover, T_h can be tuned with very fine granularity (± 1 bit), allowing precise control of the maximum allowed approximation.

Both of these are fundamental advantages of Serial T0 with respect to variable bit-width transmission, i.e., reducing the number of bits assigned to each color component of a pixel. In fact, most serial protocols, including CSI-2, do not allow data to be packed in units smaller than one byte. Others, e.g., SPI, are even more restrictive, being based on fixed-width transmitter/receiver shift registers. While effective bit-width reduction can still be obtained zeroing-out or rounding some LSBs of each pixel component (as in [12]), this solution only allows to *coarsely* tune the maximum error to different powers of 2. As shown in Section V, Serial T0 outperforms this solution in terms of quality versus power tradeoff.

The dynamic (runtime) and fine-grain tuning of the acceptable error in Serial T0 allows to switch between different "degrees" of approximation depending on external inputs, such as the state of charge of the system battery or an application-level configuration. For instance, a security camera system [22] can use a higher value of T_h when performing motion detection, as precise video information is not required in that phase. Then, it can switch to a lower T_h when the actual video is being recorded. This reconfiguration can be performed in a single clock cycle, and independently from the underlying serial protocol, thus allowing total integration with legacy peripherals without the need of hardware modifications.

B. DC Balancing

Most standard serial bus protocols (e.g., SPI, I2C, etc.) do not specify any constraint relative to the balancing of 0s and 1s in the stream, i.e., *DC balancing* [23]. Although this issue is not taken into account in the original Serial T0 algorithm, it is possible to devise a variant that does, without a significant increase in codec complexity.

The prevalence of 0s or 1s in Serial T0 codewords strongly depends on the selection of the 0-TC pattern, as discussed in Section III-B. If the $(0...0)$ pattern is used, codewords will be dominated by 0s, and vice versa.

It can be easily observed that, if these two patterns are used alternatively, Serial T0 will not worsen the DC balance of input data. On the contrary, it will tend to improve it. This solution is feasible for image sensors data since, as discussed in Section III-B, both $(0...0)$ and $(1...1)$ are uncommon patterns.

In order to implement DC balanced Serial T0, a 1-bit counter (toggle Flip Flop) must be used in order to change the used 0-TC pattern in every clock cycle. When a "real" $00..00$ or $11..11$ pattern has to be transmitted, it will be approximated with $00..01$ or $11..10$, respectively. The modifications to the encoding and decoding circuits of Figures 3 and 4 are trivial and not reported for brevity.

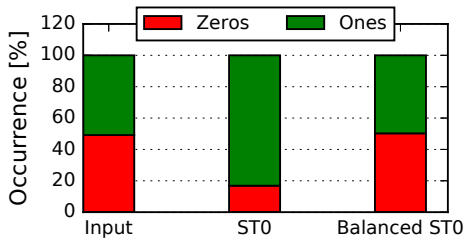


Fig. 6: Occurrence of 0s and 1s for the Lena Image. Both Serial T0 variants use $T_h = 8$.

An example of the effectiveness of this variant is shown in Figure 6, which reports the probability of occurrence of logic values 0 and 1 in the *Lena* image [24]. The leftmost bar refers to unencoded input pixels, for which DC balance is almost perfect. The central column has been obtained after Serial T0 encoding with $T_h = 8$, which yields a TC reduction of $\approx 63\%$, but with a strong bias towards the logic 1, due to the use of $(1...1)$ 0-TC pattern. The rightmost column shows the result obtained by using the DC-balanced version of Serial T0 described above; The TC reduction of the balanced version is $\approx 52\%$, now with an almost perfectly DC balancing (50.1% of logic 1s). It must be emphasized that the DC-balanced version of Serial T0 *does not* aim at improving the TC reduction of the basic encoding; conversely it will yield inferior TC reduction results, due to the larger number of inter-word transitions caused by alternations between all-zeros and all-ones patterns.

V. Experimental Results

A. Comparison with State-of-the-Art Encodings

To evaluate the effectiveness of Serial T0 as a low power encoding for image sensors, we first tested its performance in comparison to that of state-of-the-art (SOA) encodings. We

Experiment	Name	Content	Size [pixels]	Ref.
SOA Comparison	GRAZ-01	People	640x480	[25]
	CBCL Cars	Cars	128x128	[26]
	CBCL StreetScenes	Streets	1280x960	[27]
Face Recognition	Faces94	Faces	180x200	[28]
OCR	ICDAR 2003	Text	Variable	[29]

TABLE II: Input datasets.

selected three publicly available datasets of camera images as inputs, differing in image content and size, as summarized in the top part of Table II. 400 images from each set have been used for testing. Each image has been extracted as 24-bit RGB map, before simulating a serial transmission. We compare Serial T0 (and its DC balanced variant) with two SOA *approximate* serial encodings: a general-purpose encoding, Rake [4], and a solution specifically devised for image data, called LSBS [12]. Moreover, we also consider two representative *accurate* encodings: with the same rationale we select the general-purpose solution SILENT [6] and the image-specific algorithm k-LIWT [10]. For the latter encoding, the 2-LIWT variant has been selected, as it is considered the best in terms of performance versus overhead tradeoff by the authors of [10]. Consistently to the papers presenting these competitor encodings, in this phase we evaluate the reduction of the Transition Count (TC) as a proxy for power saving. Without loss of generality, we assume a simplified serial protocol that transmits data words separately from control information, on a dedicated physical wire. Notice that real protocols (e.g., SPI) belong to this category: as discussed in Section IV, for more complex standards (e.g., CSI-2) the TC reduction simply has to be weighted by the ratio between data and control bits.

Experiments have been performed under two maximum relative error constraints, i.e., $E_r = 2\%$ and $E_r = 4\%$. These constraints have been imposed by setting the Serial T0 parameter to $T_h = \lfloor E_r \cdot FS/100 \rfloor$, where the floor operator ensures that the constraint is never violated. Error parameters in competitor approximate encodings have been in a similar way, to generate a fair comparison. SILENT and 2-LIWT, being accurate, achieve a fixed TC reduction independently of the error.

Results are summarized in Figure 7. The bar charts report the mean TC reductions obtained by each encoding on the three datasets, as well as the corresponding standard deviation intervals. Serial T0 (ST0 in the figure) clearly outperforms all competitor encodings for all datasets. The balanced version of ST0 (Bal ST0) achieves the second best TC reduction, and despite the additional DC balancing feature, still outperforms all other encodings. Strikingly, the difference in TC reduction between Serial T0 and the best competitor approximate encoding is larger than 20% for *all* input sets, under the $E_r = 4\%$ condition. Also, Serial T0 and its balanced variant are the only approximate encodings able to always outperform accurate ones, regardless of the data, thus *justifying* the presence of approximations. Instead, both Rake and LSBS sometimes achieve *worse* TC reduction compared to SILENT and 2-LIWT (e.g., for Streets images), even when E_r is set to 4%.

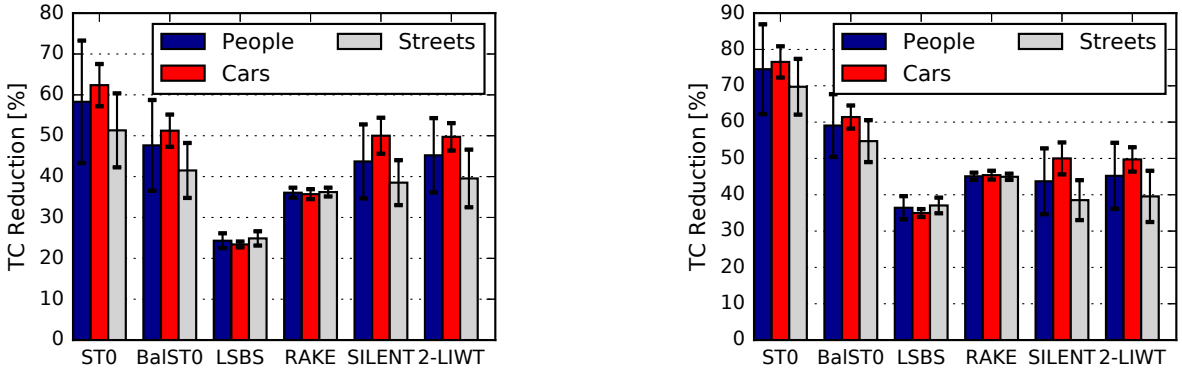
B. Evaluation using an Application-Level Metric

In a second set of experiments, we demonstrate the effectiveness of Serial T0 for two realistic applications: Optical Character Recognition (OCR) and Face detection/recognition. In particular, we assess the impact of transmitting a camera image with an approximate encoding on the performance of these two recognition algorithms. The selected faces and text datasets are reported in Table II. OCR is performed using the open-source software tesseract, currently maintained by Google [30]. Face detection uses Haar-like features cascade [31], whereas face recognition is obtained through a Local Binary Pattern Histogram (LBPH) recognizer [32]. Both algorithms are already included in the OpenCV [33] library, and have been used with default parameters. In this second set of experiments, Serial T0 is compared with Rake, which from the previous section appears to be the best competitor among approximate encodings. Moreover, we maintain the same protocol assumptions used in the previous experiment.

For both applications, we have first pruned the input dataset including only images for which recognition is correct *without* approximations. Then we have evaluated the results of recognition after a simulated ST0/Rake serial transmission, for all images in the restricted datasets. This experiment has been repeated increasing the maximum error parameter of both encodings in steps of one unit, starting from $T_h = 1$, i.e., $\approx 0.4\%$ of the full-scale. Recognition accuracy has been evaluated simply in terms of Hit Ratio (HR) for face recognition, while for OCR we used Levenshtein distance [34].

Results of this experiment are reported in Figure 8, where the horizontal axes report the maximum relative error (E_r) corresponding to the different values of T_h and the vertical axes show the corresponding TC reduction and accuracy. Under the same error constraint, Serial T0 not only achieves greater TC reductions, but also allows to preserve recognition accuracy better than Rake. This is thanks to the fact that Serial T0 only makes approximations in image regions with strong spatial correlation. Assuming 95% accuracy as a reasonable target, Serial T0 allows to reduce the transitions on the bus of about 60% for both applications. For face recognition, Rake only generates a TC reduction of $\approx 30\%$ for the same target quality. Furthermore, the Rake accuracy for the OCR application drops below 95% even with the minimum parameter value. These results confirm the effectiveness of the Serial T0 encoding in realistic scenarios.

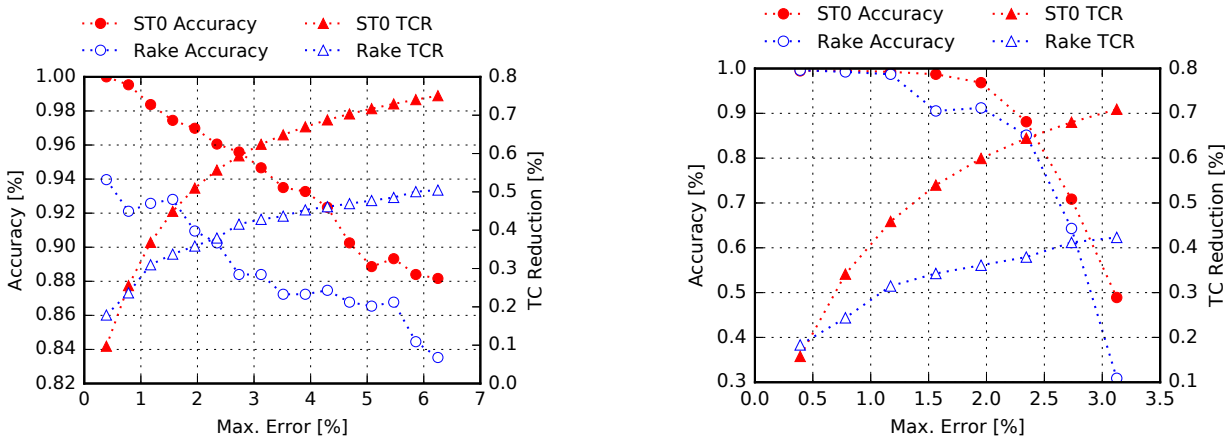
Notice that the OCR experiment had already been performed by the authors of Rake under a similar setup in [4] However, the results reported there are different from ours. The reason of the difference is twofold. First, we use slightly different input set: after pruning, our OCR dataset contains 431 images, while only 392 are used in [4]. Secondly, we assume that the results in [4] refer to a different variant of Rake from the one reported in the paper, which is the only one that we could re-implement for comparison. In any case, even considering the accuracy/TC reduction results of [4], Serial T0 would still be significantly superior to Rake, yet with much lower codec complexity.



(a) Maximum Relative Error $E_r = 2\%$

(b) Maximum Relative Error $E_r = 4\%$

Fig. 7: Transition Count reduction of Serial T0 and competitor encodings for camera images.



(a) Optical Character Recognition (OCR)

(b) Face Recognition

Fig. 8: TC reduction versus accuracy of Serial T0 and Rake for two realistic classification applications.

C. Accurate power saving analysis

The last experiment concerns the assessment of the actual energy reduction considering the overheads due to the codec. As an example, we assume that the goal is to design an interconnection able to transmit 1024x768 24-bit RGB images at a rate of 30Hz, using three LVDS lanes, i.e., one per color channel. For simplicity, we assume that no control information is transmitted on data wires, and that the three lanes are equipped with independent 8-bit Serial T0 encoders. This allows us to just analyze *one* of the lanes in the following, as each of them can be considered as an independent serial connection. The throughput requirement corresponds to a minimum transmission frequency per lane of $f_{clk} = 1024 \cdot 768 \cdot 8 \cdot 30 \simeq 189$MHz; to ease calculations, we assume in the following $f_{clk} = 200$MHz (a value within the range indicated in the LVDS specifications). As realistic model of the physical connection layer, we use a CAT6 twisted pair [35], which can support a maximum frequency of 250MHz, and has a nominal capacitance per unit length of $C_{LEN} = 46pF/m$. Moreover, we assume $V_{swing} = 1.4V - 1.0V = 400mV$, in accordance to the LVDS specifications [18].

To evaluate codec costs, the circuits of Figure 3 and 4 have been described in VHDL and synthesized using Synopsys Design Compiler K-2015.06, targeting a 45nm CMOS standard cell library from ST Microelectronics, with a clock frequency of 200MHz, under the assumption that the entire transmitter (i.e., codec, serializer and driver) works synchronously. Obviously, since the codec processes a word per clock cycle, we assume that the circuit is power managed (with clock gating etc.) in the remaining 7 cycles, hence not consuming energy, in first approximation. The power consumption of the synthesized netlists has been estimated with Synopsys PrimeTime J-2014.12, using averaged switching activity. Synthesis results are reported in Table III.

Circuit	Area [μm^2]	Power [W]	E^w_{HW} [J]
Encoder	304.82	$4.59 \cdot 10^{-5}$	$2.295 \cdot 10^{-13}$
Decoder	73.38	$1.66 \cdot 10^{-5}$	$8.3 \cdot 10^{-14}$

TABLE III: Codec Synthesis Results @ $f_{clk} = 200$ MHz.

Table IV summarizes the results of an accurate energy consumption evaluation performed on the same data of Section V-A. Line parameters are combined with the switching probability α as in (1) to derive the average line power consumption per unit length. Combining this value with frequency and bit-width information yields the energy per unit

Dataset	Input		Serial T0 - 2%				Serial T0 - 4%			
	α	$E_{IN}^{l,w}$ [pJ]	α	$E_{ST0}^{l,w}$ [pJ]	l_{BE} [cm]	E_S [%]	α	$E_{ST0}^{l,w}$ [pJ]	l_{BE} [cm]	E_S [%]
People	0.51	30.0	0.21	12.5	1.79	**32.2**	0.13	7.6	1.40	**48.5**
Cars	0.53	31.5	0.20	11.8	1.59	**37.6**	0.12	7.4	1.30	**51.8**
Streets	0.50	29.4	0.24	14.3	2.08	**24.6**	0.15	8.9	1.53	**43.1**

TABLE IV: Break-even length analysis and total energy saving evaluation for a 4cm CAT6 cable.

length consumed by the line to transmit one word $E^{l,w}$. The table reports both α and $E^{l,w}$ for unencoded data, as well as after Serial T0 encoding, with the two values of T_h used in Section V-A. Serial T0 reduces line energy by reducing α and spending an energy overhead due to the codec E_{HW}^{w} (Table III). The impact of this overhead is clearly smaller if the line capacitance increases, i.e., for longer cables. Consumption due to line drivers, I/O pads and serializer is constant regardless of the encoding, hence it can be removed from the analysis.

We define as *break-even length* (l_{BE}) the minimum line length for which the codec overheads are amortized by the TC reduction on the line. Values of l_{BE} for the different inputs and values of T_h are reported in Table IV, confirming the effectiveness of Serial T0. In fact, l_{BE} is in the order of 1.5 cm, on average, whereas off-chip camera connections made with twisted pairs are normally several centimeter long. Column E_S reports the total energy savings (considering all overheads) for a 4 cm long cable. In the best case, energy can be reduced of more than 50%. Notice that in this experiment we could not compare Serial T0 with its main competitor among approximate encodings (Rake), as the authors of [4] did not propose an implementation of the codec. However, it is intuitive from the algorithm that Rake, which implements a two-pass bit-by-bit swipe of each data word, is significantly more complex than Serial T0 to implement in hardware.

VI. Conclusions

Consumption due to long off-chip serial interconnections can be a significant contributor to the total energy breakdown of an embedded system for sensor applications. The proposed Serial T0 encoding exploits the spatial correlation of image pixels to address this issue for image sensors such as embedded cameras. By doing so, it obtains a substantial reduction of the energy consumption on the line, despite a very simple codec implementation. Moreover, it can be easily integrated with existing standards and legacy peripherals.

Experiments show that Serial T0 is superior to all state-of-the-art encodings for serial buses, both when analyzed in terms of pure transition count reduction, and when used in the context of a complete application.

References

[1] F. Vahid and T. Givargis, *Embedded System Design: A Unified Hardware/Software Introduction*, 1st ed. John Wiley & Sons, Inc., 2001.

[2] H. Hatamkhani et al, *Power-centric design of high-speed I/Os*, in *ACM/EDAC/IEEE DAC*, 2006, pp. 867-782.

[3] N. Ickes et al, "A 10-pj/instruction, 4-mips micropower dsp for sensor applications," in *IEEE A-SSCC*, 2008, pp. 289–292.

[4] P. Stanley-Marbell and M. Rinard, "Reducing serial i/o power in error-tolerant applications by efficient lossy encoding," in *ACM/EDAC/IEEE DAC*, 2016, pp. 62:1–62:6.

[5] D. Jahier Pagliari, E. Macii, and M. Poncino, "Serial t0: Approximate bus encoding for energy-efficient transmission of sensor signals," in *ACM/EDAC/IEEE DAC*, 2016, pp. 14:1–14:6.

[6] K. Lee, S.-J. Lee, and H.-J. Yoo, "Silent: serialized low energy transmission coding for on-chip interconnection networks," in *IEEE/ACM ICCAD*, 2004, pp. 448–451.

[7] X. Ren et al, "Adaptive low-power transmission coding for serial links in network-on-chip," *Procedia Engineering*, vol. 29, pp. 1618–1624, 2012.

[8] S. Ghosh et al, "Data correlation aware serial encoding for low switching power on-chip communication," in *IEEE ISVLSI*, 2014, pp. 124–129.

[9] J. Zeng et al, "Transition inversion coding with parity check for off-chip serial transmission," in *IEEE ICECS*, 2014, pp. 634–637.

[10] S. Salerno et al, "Limited intra-word transition codes: an energy-efficient bus encoding for lcd display interfaces," in *ISLPED*, 2004, pp. 206–211.

[11] M. Saneei et al, "Serial Bus Encoding for Low Power Application," in *International Symposium on SoC*, 2006, pp. 1–4.

[12] M. Poncino and E. Macii, "Low-energy rgb color approximation for digital lcd interfaces," *IEEE Trans. on Consum. Electron.*, vol. 52, no. 3, pp. 1004–1012, Aug. 2006.

[13] V. Chippa et al, "Analysis and characterization of inherent application resilience for approximate computing," in *ACM/EDAC/IEEE DAC*, May 2013, pp. 1–9.

[14] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *IEEE ETS*, 2013, pp. 1–6.

[15] L. Benini et al, "Asymptotic zero-transition activity encoding for address busses in low-power microprocessor-based systems," in *GLSVLSI*, 1997, pp. 77–82.

[16] *Camera Interface Specifications*, MIPI Alliance, 2005.

[17] Y. C. Lu et al, "Low power multi-lane mipi csi-2 receiver design and hardware implementations," in *IEEE ISCE*, 2013, pp. 199–200.

[18] *Physical Layer Specifications*, MIPI Alliance, 2009. Url: http://mipi.org/specifications/physical-layer

[19] NXP, *I2C-bus specification and user manual*, NXP, 2014.

[20] Motorola, *MC68HC11 Reference Manual*, 1989.

[21] ISO, *ISO 11898-1:2015, Road vehicles – Controller area network (CAN)*, International Organization for Standardization, 2015.

[22] S.-M. Sohn, et al, "A cmos image sensor (cis) architecture with low power motion detection for portable security camera applications," *IEEE Trans. on Consum. Electron.*, vol. 49, no. 4, pp. 1227–1233, Nov 2003.

[23] J. G. Webster and H. Eren, *Measurement, Instrumentation, and Sensors Handbook*, 2nd ed. CRC Press, 2014.

[24] Kodak, *Image Database*, http://r0k.us/graphics/kodak.

[25] A. Opelt and A. Pinz, *GRAZ-01 Database*, Graz, University of Technology, Austria, 2003.

[26] *CBCL Car Database*, MIT Center for Biological and Computational Learning (CBCL), 2000.

[27] *CBCL StreetScenes*, MIT Center for Biological and Computational Learning (CBCL), 2007.

[28] L. Spacek, *Face Recognition Data*, University of Essex, UK, 2007.

[29] S. Lucas, *Robust Word Recognition Dataset*, ICDAR, 2003.

[30] R. Smith, "An overview of the tesseract ocr engine," in *IEEE ICDAR*, 2007, pp. 629–633.

[31] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE CVPR*, 2001, pp. I–511–I–518.

[32] T. Ojala, M. Pietikinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51 – 59, 1996.

[33] *Open Source Computer Vision Library*. Url: http://opencv.org/

[34] G. Navarro, "A guided tour to approximate string matching," *ACM Comput. Surv.*, vol. 33, no. 1, pp. 31–88, 2001.

[35] *ISO/IEC 11801 - Information technology Generic cabling for customer premises*, 2nd ed., ISO/IEC, 2010.